Bill Weinberg

Senior director of open-source strategy at Black Duck Software and GENIVI contributor

Protecting the connected car from hacking could be achieved by using open-source software platforms

Over the last 24 months, open-source software has found itself in the crosshairs of security hackers, and, by consequence, in the media spotlight as well. Of the 10,000-plus vulnerabilities catalogued in 2014, more than 60% surfaced in proprietary code. However, that leaves 4,000 potentially exploitable flaws in open-source, and in 2015 threat discovery is running at an even greater pace.

Recent vulnerabilities notwithstanding, open-source is still regarded as more secure by both enterprise and embedded developers, for its transparency, code quality, and the community purview and processes that govern and manage it.

Through the GENIVI Alliance and its members, the in-vehicle infotainment (IVI) ecosystem increasingly uses open-source software, especially embedded Linux, networking code, and graphical and multimedia frameworks in IVI equipment. Consequently, security is of keen and critical interest to OEMs, Tier One suppliers and other participants.

The interest falls into three key areas: firstly, security in the supply chain. The automotive industry in general, and the IVI segment in particular, is a multi-sourced ecosystem. OEMs today increasingly integrate their own legacy code with code from Tier One suppliers which themselves obtain software from open-source community projects, commercial open-source vendors, semiconductor and systems suppliers, and from other third parties.

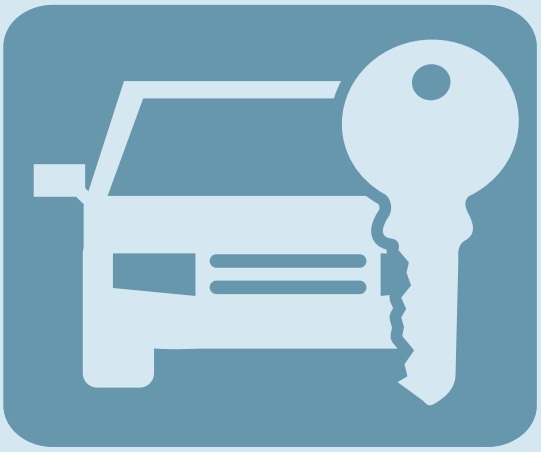
Any of these suppliers may also obtain code from and exhibit dependencies upon still other open-source (and proprietary) software technologies. The result is a complicated supply chain of interdependencies that must be tracked and carefully managed.

Gone is the day when suppliers could deliver source code, shrink-wrapped software or software embedded in hardware subsystems 'as is', or ship their wares accompanied by unvalidated bills of material (BOMs). Acceptance today is founded on detailed information about the origin of each software component, its provenance and licensing, and other assurances that the BOM received is both comprehensive and accurate.

Today's dynamic security landscape adds a new requirement to this already complex supply chain: security. In particular, specific knowledge of delivered versions of open-source (and other) software and known vulnerabilities logged against each component (if any) must be managed.

The second key area is common security architectures. Recently, a number of highly visible exploitations of IVI systems and, more perilously, hacks of CAN-based control systems of popular vehicles have occurred. These exploits have led industry experts to question presumptions about inter-system isolation and 'unhackability' of increasingly connected cars, and to critique complacence and unwarranted confidence in existing automotive system architectures.

These exploits and an escalating threat level constitute a wake-up call for the whole automotive ecosystem to re-evaluate software and hardware architectures. Rather than each OEM and/or supplier working in isolation, the industry needs to work together (in communities such as GENIVI, AUTOSAR and others) to examine assumptions about automotive security, and to upgrade and standardise software architectures to equal (and surpass) security levels found in other IT disciplines.



"Systems will only exhibit necessary resilience when security best practices become part of developer mindsets"

Finally, best practices for security in the automotive application lifecycle, securing the supply chain and architecting for security will be most effective when vulnerability checking and architectural guidelines are tightly integrated into the software development lifecycle.

Security is not something that should be added 'after the fact'. Automotive systems will only exhibit necessary resilience when security best practices become part of developer mindsets and of the workflows and tooling they use to design, create, test, integrate and deploy software in sub-systems and products they supply.

These practices include following modern software engineering guidelines for code quality and security in creating code. Incoming/integrated open-source code should be scanned for deprecated versions, version proliferation and vulnerabilities. Code analysis tools should be employed to eliminate common exploitable bugs and faults. There is also a need for long-term monitoring for vulnerabilities and other security issues in vehicle software stacks. These practices and requirements should be propagated upstream to suppliers and downstream to customers.

Security in the vehicle will increase through appropriate supply-chain management, the development of common security architectures, and the application of best practices for security during the software development lifecycle. ■